EXHIBIT 12

DOCKET NO: 0100157-00240

IN THE UNITED STATES PATENT AND TRADEMARK OFFICE

PATENT: 8,001,096

INVENTORS: DAVID A. FARBER
AND RONALD D. LACHMAN

FILED: OCT. 31, 2007 ISSUED: AUG. 16, 2011

TITLE: COMPUTER FILE SYSTEM
USING CONTENT-DEPENDENT
FILE IDENTIFIERS

Mail Stop PATENT BOARD
Patent Trial and Appeal Board
U.S. Patent & Trademark Office
P.O. Box 1450
Alexandria, VA 22313-1450

PETITION FOR *INTER PARTES* REVIEW OF U.S. PATENT NO. 8,001,096 UNDER 35 U.S.C. § 312 AND 37 C.F.R. § 42.104

TABLE OF CONTENTS

I. MANDATORY NOTICES

A. Real Party-In-Interest

EMC Corporation ("Petitioner") is the real party-in-interest.

B. Related Matters

The '096 patent is one of an extensive patent family of continuation and divisional applications. Exhibit 1008 shows the patent family, with patents in red and blue including the '096 patent being asserted in the litigation *PersonalWeb Technologies LLC v. EMC Corporation and VMware, Inc.* (No. 6:11-cv-00660-LED) (E.D. Tex.), served on December 16, 2011.

Petitioner is also seeking *Inter Partes* Review of related U.S. Patents Nos. 5,978,791, 6,415,280, 7,945,539, 7,945,544, and 7,949,662, and requests that they be assigned to the same Board for administrative efficiency. Moreover, there are several continuing applications related to this family that remain pending (shown on Ex. 1008 in green). Because they share a common disclosure with the '096 patent, these applications may be used as a basis to present patentably indistinct claims that may issue prior to the determination of the PTAB in this or related Inter Partes Reviews. The issuance of indistinct claims is at least inconsistent with Rule 37 C.F.R. 42.73(d)(ii) and would be an "end-around" the reasonable number of substitute claims that are permitted in an IPR proceeding. Petitioner respectfully requests that the PTAB suspend from further prosecution, *sua sponte*, the applications in this related family, including the applications shown on Exhibit 1008 in green and any further applications that may be filed that depend from this family of patents. If the PTAB determines that that suspension should be requested by written motion, permission to file such a motion is requested at this time.

C. Counsel

Lead Counsel: Peter M. Dichiara (Registration No. 38,005)

Backup Counsel: David L. Cavanaugh (Registration No. 36,476)

Petitioners will request authorization to file a motion for Cynthia Vreeland to appear *pro hac vice*. Ms. Vreeland has more than 20 years litigation experience, and has worked with Petitioner EMC on IP litigation matters for more than 10 years. As such, Ms. Vreeland has experience and established familiarity with the technology at issue in the case. Petitioners intend to file a motion seeking admission of Ms. Vreeland to appear *pro hac vice* when authorized to do so.

D. Service Information

Email: Peter Dichiara, peter.dichiara@wilmerhale.com

Post and Hand Delivery: WilmerHale, 60 State St., Boston MA 02109

Telephone: 617-526-6466 Facsimile: 617-526-5000

E. Certification of Grounds for Standing

Petitioner certifies pursuant to Rule 42.104(a) that the patent for which review is sought is available for *inter partes* review and that Petitioner is not barred or estopped from requesting an *inter partes* review challenging the patent claims on the grounds identified in this Petition. Service of the complaint in *PersonalWeb Technologies LLC v. EMC Corporation and VMware, Inc.* (No. 6:11-cv-00660-LED) (E.D. Tex.) occurred on December 16, 2011. The one year time period for filing an *inter partes* review petition occurred on Sunday, December 16, 2012. This petition is timely filed the next business day, December 17, 2012. *See* 35 U.S.C. § 21.

II. OVERVIEW OF CHALLENGE AND RELIEF REQUESTED

A. Prior Art Patents and Printed Publications

Pursuant to Rules 42.22(a)(1) and 42.104 (b)(1)-(2), Petitioner challenges claims 1, 2, 81, and 83 of U.S. Patent No. 8,001,096 ("the '096 patent", Ex. 1001) as anticipated by or unpatentable in view of the following patents and printed publications:

1. S. Browne et al., "Location-Independent Naming for Virtual Distributed Software Repositories," University of Tennessee

Technical Report CS-95-278 (Feb. 1995) ("Browne ", Ex. 1002).[1]

2. Albert Langer, "Re: dl/describe (File descriptions)," article <1991Aug7.225159.786@newshost.anu.edu.au> in Usenet newsgroups "alt.sources.d" and "comp.archives.admin" (August 7, 1991) ("Langer", Ex. 1003)[2]

[1] The Browne February 1995 publication qualifies as prior art under 35 U.S.C. § 102(a), and is used in this petition because it includes illustrations which facilitate explanation of the grounds of the invalidity. Petitioner also has attached as exhibits and included in its claim charts two earlier versions of this publication – S. Browne et al., "Location-Independent Naming for Virtual Distributed Software Repositories," http://www.netlib.org/utk/papers/lifn/main.html (Nov. 11, 1994) (Ex. 1006); and K. Moore et al., "An Architecture for Bulk File Distribution," Network Working Group Internet Draft (July 27, 1994) (Ex. 1007). As Dr. Clark confirms in his declaration, the relevant disclosures are substantially the same. If the Patent Owner attempts to claim an earlier priority date of the challenged claims, Petitioner may rely on the earlier publications for invalidity, alone or in combination with the other references cited in this petition.

[2] Langer was made available on the "alt.sources.d" and "comp.archives.admin" newsgroup distribution lists on August 7, 1991. Both newsgroups were widely disseminated and readily accessible to the relevant technical community.

3. Kantor, "The Frederick W. Kantor Contents-Signature System Version 1.22," FWKCS122.REF (August 10, 1993) ("Kantor", Ex. 1004).[3]

4. M. Satyanarayanan et al., "Coda: A Highly Available File System for a Distributed Workstation Environment," IEEE Transactions on Computers, vol. 39, no. 4 (April 1990) ("Satyanarayanan II," Ex. 1028.)

Specifically, the "alt.sources.d" newsgroup was devoted to technical discussions relating to the "alt.sources" source code repository. The "comp.archives.admin" newsgroup hosted discussions relating to computer archive administration. Therefore, an interested person would have been able to readily locate Langer among postings related to those subjects, both of which are in the same technical field as the '096 patent.

[3] Kantor's FWKCS user manual has been publicly and freely available continuously since August 1993. Kantor distributed the user manual with the FWKCS program as shareware and posted it online to electronic Bulletin Board Systems including "The Invention Factory" and "Channel 1" for an extended period of time, where it could be downloaded by anyone. As such the document was accessible to others in the relevant community of BBS users and system operators. (*See* Kantor at 3; *see also* 158-59; Ex. 1004.)

B. There is a Reasonable Likelihood that at least One Claim of the '096 Patent is Unpatentable Under 35 U.S.C. §§ 102, 103

Section VI below explains how the above-cited patents and printed publications create a reasonable likelihood that Petitioner will prevail with at least one of the challenged claims. *See* 35 U.S.C. § 314(a). Indeed, that section together with the claim charts (Exs. 1029, 1030, 1036) and the Declaration of Dr. Douglas Clark, a Professor of Computer Science at Princeton University ("Clark Decl."; Ex. 1009), demonstrate that all of the challenged claims are anticipated by, or unpatentable in view of, each of these references.

C. Relief Requested

Petitioner requests cancellation of claims 1, 2, 81 and 83, the challenged claims, as unpatentable under 35 U.S.C. §§ 102 and 103.

III. Claim Construction

The claim terms should be given their "broadest reasonable construction in light of the specification." 37 C.F.R. § 42.100(b).

The claim terms can be understood by their plain and ordinary meanings except where construed in the specification. The specification includes the following constructions relevant to the challenged claims:

Claim Term	Construction
"data" and "data	"as used herein refer to sequences of bits. Thus a data item may

Claim Term	**Construction**
item"	be the contents of a file, a portion of a file, a page in memory, an object in an object-oriented program, a digital message, a digital scanned image, a part of a video or audio signal, or any other entity which can be represented by a sequence of bits" ('096 patent, col. 2, ll. 16-21, *see also* col. 2, ll. 26-38 (indicating "data items" can include "files, directories, records in the database, objects in object-oriented programming, locations in memory or on a physical device or the like"); '096 patent, claim 13 (indicating "data item" can include, in addition to the items above, "a software product [or] a portion of a software product".); Ex. 1001.)
"file system"	"a collection of directories. A directory is a collection of named files – both data files and other directory files" ('096 patent, col. 5, ll. 61-63; Ex. 1001.)
"file"	"a named data item which is either a data file (which may be simple or compound) or a directory file. A simple file consists of a data segment. A compound file consists of a sequence of data segments. A data segment is a fixed sequence of bytes" ('096 patent, col. 5, l. 64 – col. 6, l. 1; Ex. 1001.)
"location"	"with respect to a data processing system, refers to any of a particular processor in the system, a memory of a particular processor, a storage device, a removable storage medium (such as a floppy disk or compact disk), or any other physical location

Claim Term	**Construction**
	in the system" ('096 patent, col. 6, ll. 12-17; Ex. 1001.)
"True Name, data identity, and data identifier"	"refer to the substantially unique data identifier for a particular item" ('096 patent, col. 6, ll. 20-22; *see also* col. 13, l. 31 – col. 14, l. 2 (describing mechanism for calculating True Name using MD hash function); Ex. 1001.)

IV. OVERVIEW OF THE '096 PATENT

A. Brief Description

The '096 patent is directed to data storage systems that use "substantially unique data identifiers" – based on all the data in a data item and only the data in the data item – to identify and access data items. (*See, e.g.*, '096 patent, Title, Abstract, and col. 1, ll. 44-48; Ex. 1001.) The patent uses these identifiers to perform basic file management functions, such as requesting and obtaining computer files or other data items, and eliminating unwanted duplicate copies of data items—admittedly old problems. (*See, e.g.*, '096 patent, Background of the Invention, col. 3, ll. 4-15; Ex. 1001.)

According to the patent, prior art systems identified data items based on their location or address within the data processing system. ('096 patent, col. 1, ll. 53-60; Ex. 1001.) For example, files were often identified by their context or "pathname," that is, information specifying a path through the computer directories to the particular file (*e.g.*, C:\My Documents\Law School\1L\TortsOutline.txt). ('096 patent, col. 1, l. 65 – col. 2, l. 5; Ex. 1001.) The patent contends that all prior art systems operated in this manner: "In ***all*** of the prior data processing systems, the names or identifiers provided to identify data items. . . are ***always*** defined relative to a specific context," and "there is ***no*** direct relationship between the data names and the data item." ('096 patent, col. 2. ll. 26-31, ll. 39-40 (emphasis added); Ex. 1001.)

According to the patent, this prior art practice of identifying a data item by its context or pathname resulted in certain shortcomings. For example, with pathname identification, the same data name may refer to different data items, or conversely, two different data names may refer to the same data item. ('096 patent, col. 2, ll. 39-43; Ex. 1001.) Moreover, because there is no correlation between the contents of a data item and its pathname, there is no *a priori* way to confirm that the data item is in fact the one named by the pathname. ('096 patent, col. 2, ll. 44-47; Ex. 1001.) Furthermore, context or pathname identification may more easily result in the creation of unwanted duplicate data items, e.g., multiple copies of a file on a file server.[4] (‘096 patent, col. 3, ll. 4-15; Ex. 1001.)

The ‘096 patent purports to address these shortcomings. (‘096 patent, col. 3, ll. 30-48; Ex. 1001.) It suggests that “it is therefore desirable to have a mechanism . . . to determine a common and substantially unique identifier for a data item, using only the data in the data item and not relying on any sort of context.” (‘096 patent, col. 3, ll. 31-35; Ex. 1001.) To do so, the ‘096 patent provides data identifiers that “depend[] on all of the data in the data item and only on the data in the data item.” (‘096 patent, col. 3, ll. 52-55; *see also* Field of the Invention (“This invention relates to data processing systems… wherein data items are identified by substantially unique identifiers which depend on all of the data in the data items and only on the data in the data items”), col. 1, ll. 44-48; Ex. 1001.) The preferred embodiments use either of the well-known MD5 or SHA message digest hash functions[5] to calculate a substantially unique identifier from the contents of the

[4] For example, Alice and Bob both download the same copy of the James Bond movie *Goldfinger*. Alice saves her copy at “C:\Movies\Bond\Goldfinger.mov”, and Bob saves his copy at “C:\Videos\007\Bond-Goldfinger.mov”.

[5] A message digest or hash function transforms of a piece of data into a much

data item. (‘096 patent, col. 12, l. 18 – col. 14, l. 2; Ex. 1001.) The system first computes the 16-byte (128-bit) message digest of the data item and then appends the size of the data item to produce a 160-bit identifier. (‘096 patent, Fig. 10A and col. 13, ll. 31-42; Ex. 1001.) The patent calls these context- or location-independent, content-based identifiers a “True Name” – a phrase admittedly “coined by the inventors.” (U.S. Patent No. 6,415,280 Prosecution History, Response (Aug. 22, 2001), at 22; Ex. 1019.)

If a data item is large, it may include multiple components or “segments,” representing the larger “compound data item.” (‘096 patent, col. 13, ll. 43-53; Ex. 1001.) A True Name identifier can be computed for each of the segments based on a hash of the contents of the segment. (‘096 patent, col. 13, ll. 31-53; Ex. 1001.) Together, these segment identifiers form an “indirect block.” (‘096 patent, col. 13, ll. 53-57; Ex. 1001.) A True Name identifier is then computed for the compound data item as a whole based on a hash of the contents of the indirect

shorter form by performing mathematical operations on its content. (*See, e.g.*, D. Banisar et al., The Third CPSR Cryptography and Privacy Conference at 509 (1993) (describing a message digest function); Ex. 1010.) The ‘096 patent admits that message digest functions were known. (‘096 patent, col. 12, ll. 40-49; Ex. 1001.)

block (*i.e.*, a hash of the segment hashes). (‘096 patent, col. 13, ll. 55-61; Ex. 1001.)

With these identifiers, the patent asserts, “data items can be accessed by reference to their identities (True Names) independent of their present location.” (‘096 patent, col. 33, ll. 28-30; *see also* col. 33, ll. 49-51; Ex. 1001.) The actual data item corresponding to these location-independent identifiers may reside anywhere, e.g., locally, remotely, offline. (‘096 patent, col. 33, ll. 30-38; Ex. 1001.) “Thus, the identity of a data item is independent of its name, origin, location, address, or other information not derivable directly from the data, and depends only on the data itself.” (‘096 patent, col. 3, ll. 55-58; Ex. 1001.)

In the preferred embodiments, the substantially unique identifiers are used to “augment” standard file management functions of an existing operating system. (‘096 patent, col. 6, ll. 25-32; Ex. 1001.) For example, a local directory extensions (LDE) table[6] is indexed by a pathname or contextual name of a file and also includes True Names for most files. (‘096 patent, col. 8, ll. 28-36; Ex. 1001.) A

[6] According to the patent, a LDE table is a data structure which provides information about files and directories in the system and includes information in addition to that provided by the native file system. (‘096 patent, col. 8, ll. 28-36; Ex. 1001.)

True File registry (TFR) lists True Names, and stores "location, dependency, and migration information about True Files." ('096 patent, col. 8, ll. 37-39, 42-44; Ex. 1001.) True Files are identified in the True File registry by their True Names, and can be looked up in the registry by their True Names. ('096 patent, col. 8, ll. 40-42; col. 23, ll. 25-26; Ex. 1001.) This look-up provides, for each True Name, a list of the locations, such as file servers, where the corresponding file is stored. ('096 patent, col. 33, ll. 36-38; *see also* col. 15, ll. 44-46; Ex. 1001.)

The system also includes a "Mirror True File" background mechanism "to mirror (make copies) of the True File available elsewhere in the system." ('096 patent, col. 33, ll. 4-8; Ex. 1001.) "In operation data items can be accessed by reference to their identities (True Names) independent of their present location. The actual data item or True File corresponding to a given data identifier or True Name may reside anywhere in the system (that is, locally, remotely, offline, etc)." ('096 patent, col. 33, ll. 28-32; Ex. 1001.) If a data item is not present locally, the True File registry may be used to determine the location(s) of copies of the True File corresponding to a given True Name. ('096 patent, col. 33, ll. 34-38; Ex. 1001.)

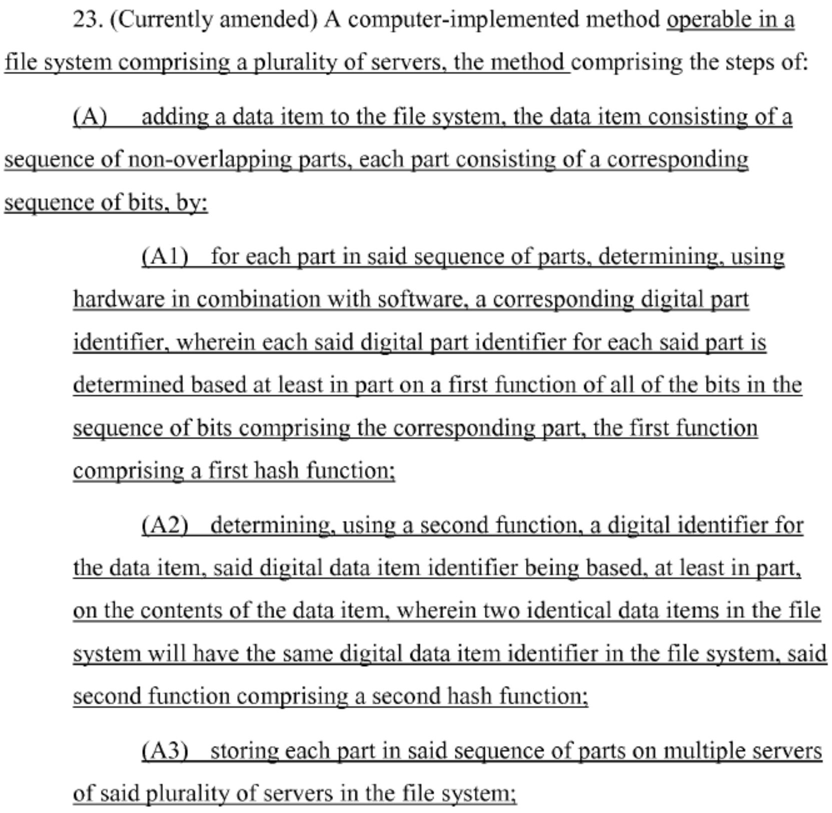
When opening or reading a file, the "Read File" mechanism "is aware of compound files and indirect blocks, and it uses [other] mechanisms to make sure that component segments are locally available. . . . When [a compound file] is opened only its indirect block is copied. When the corresponding file is read, the required component segments are realized and therefore copied." ('096 patent, col. 33, ll. 17-27; Ex. 1001.)

B. The Prosecution History of the '096 Patent

The '096 patent is based on an application that was originally filed on April 11, 1995. The application was filed on October 31, 2007 with 22 claims (Application as filed; Ex. 1024), but these claims were canceled in a preliminary amendment, at which time new claims 23-60 were introduced. (Preliminary Amendment, Apr. 12, 2010, at 12; Ex. 1044**)**.

All claims were rejected as anticipated by Cahill (U.S. Pat. No. 5,678,046) or rendered obvious by Cahill in view of Dyson (U.S. Pat. No. 5,050,212). (Office Action, June 4, 2010, at 2 and 14; Ex. 1025.)

In response, applicants interviewed the case on September 23, 2010 and the claims were amended considerably. (Response, Nov. 23, 2010; Ex. 1026.) For example, claim 23 was amended as follows:



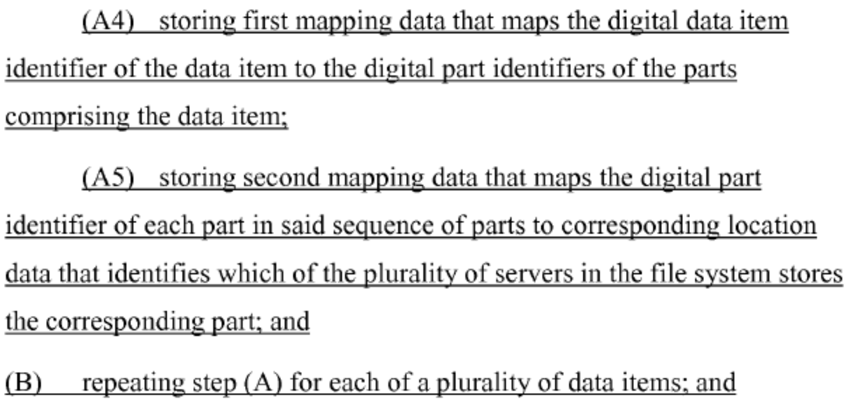
23. (Currently amended) A computer-implemented method operable in a file system comprising a plurality of servers, the method comprising the steps of:

(A) adding a data item to the file system, the data item consisting of a sequence of non-overlapping parts, each part consisting of a corresponding sequence of bits, by:

(A1) for each part in said sequence of parts, determining, using hardware in combination with software, a corresponding digital part identifier, wherein each said digital part identifier for each said part is determined based at least in part on a first function of all of the bits in the sequence of bits comprising the corresponding part, the first function comprising a first hash function;

(A2) determining, using a second function, a digital identifier for the data item, said digital data item identifier being based, at least in part, on the contents of the data item, wherein two identical data items in the file system will have the same digital data item identifier in the file system, said second function comprising a second hash function;

(A3) storing each part in said sequence of parts on multiple servers of said plurality of servers in the file system;



(A4) storing first mapping data that maps the digital data item identifier of the data item to the digital part identifiers of the parts comprising the data item;

(A5) storing second mapping data that maps the digital part identifier of each part in said sequence of parts to corresponding location data that identifies which of the plurality of servers in the file system stores the corresponding part; and

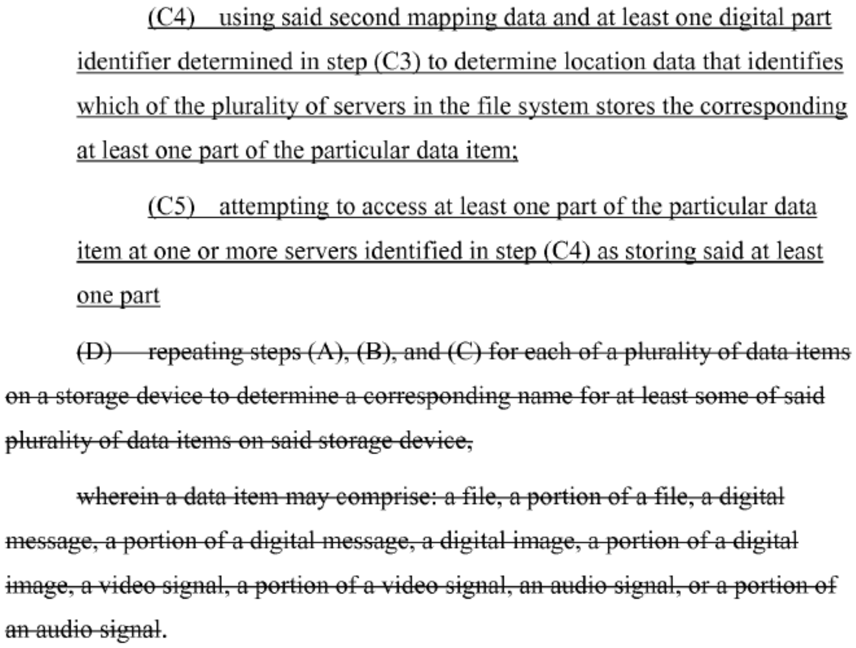
(B) repeating step (A) for each of a plurality of data items; and

(C) attempting to access a particular data item in the file system by:

(C1) [[(A)]] obtaining a particular ~~content-dependent~~ digital data item identifier of the [[a]] particular data item, said particular digital data item identifier of said particular data item being included in an attempt to access said particular data item in said file system ~~based, at least in part, on at least some data comprising the particular data item~~;

(~~B~~C2) attempting to match, using [[by]] hardware in combination with software, ~~attempting to match~~ said particular digital data item identifier of said particular data item with a digital data item identifier in said first mapping data ~~a database, said database comprising a plurality of digital identifiers, each of said digital identifiers in said database corresponding to at least one data item of a plurality of data items, wherein said database maps each said digital identifier in said database to information relating to a corresponding data item, each of said digital identifiers in said database being based, at least in part, on at least some data in a corresponding data item, wherein the information corresponding to each data item in the database includes a name for that data item~~; and

(C3) based at least in part on said attempting to match in step (C2), when said particular digital data item identifier obtained in step (C1) corresponds to an identifier in said first mapping data, using said first mapping data to determine a digital part identifier of each part comprising the particular data item; ~~(B), determining, using said database, information corresponding said particular data item, said information including at least a name for the particular data item; and~~



(C4) using said second mapping data and at least one digital part identifier determined in step (C3) to determine location data that identifies which of the plurality of servers in the file system stores the corresponding at least one part of the particular data item;

(C5) attempting to access at least one part of the particular data item at one or more servers identified in step (C4) as storing said at least one part

~~(D) repeating steps (A), (B), and (C) for each of a plurality of data items on a storage device to determine a corresponding name for at least some of said plurality of data items on said storage device,~~

~~wherein a data item may comprise: a file, a portion of a file, a digital message, a portion of a digital message, a digital image, a portion of a digital image, a video signal, a portion of a video signal, an audio signal, or a portion of an audio signal~~.

(*Id*. at 3-5.) Applicants also added over 60 new claims. (*Id*. at 18-34.) Over 30 more new claims were later added in a supplemental amendment. (Suppl. Resp., Dec. 26, 2010, at 40-49; Ex. 1027.) The claims were subsequently allowed without comment, with claim 23 renumbered to challenged claim 1. (Notice of Allowance, Apr. 22, 2011, at 6; Ex. 1045).

V. THE CHALLENGED CLAIMS ARE UNPATENTABLE

A. There is Nothing New About Replicating Data Items and Using Content-based data Identifiers to Access Data

Claim 1 of the ‘096 patent is reproduced below:

1. A computer-implemented method operable in a file system comprising a plurality of servers, the method comprising the steps of:

(A) adding a data item to the file system, the data item consisting of a sequence of non-overlapping parts, each part consisting of a corresponding sequence of bits, by:

> (A1) for each part in said sequence of parts, determining, using hardware in combination with software, a corresponding digital part identifier, wherein each said digital part identifier for each said part is determined based at least in part on a first function of all of the bits in the sequence of bits comprising the corresponding part, the first function comprising a first hash function;
>
> (A2) determining, using a second function, a digital identifier for the data item, said digital data item identifier being based, at least in part, on the contents of the data item, wherein two identical data items in the file system will have the same digital data item identifier in the file system, said second function comprising a second hash function;
>
> (A3) storing each part in said sequence of parts on multiple servers of said plurality of servers in the file system;
>
> (A4) storing first mapping data that maps the digital data item identifier of the data item to the digital part identifiers of the parts comprising the data item;

(A5) storing second mapping data that maps the digital part identifier of each part in said sequence of parts to corresponding location data that identifies which of the plurality of servers in the file system stores the corresponding part; and

(B) repeating step (A) for each of a plurality of data items; and

(C) attempting to access a particular data item in the file system by:

(C1) obtaining a particular digital data item identifier of the particular data item, said particular digital data item identifier of said particular data item being included in an attempt to access said particular data item in said file system;

(C2) attempting to match, using hardware in combination with software, said particular digital data item identifier of said particular data item with a digital data item identifier in said first mapping data; and

(C3) based at least in part on said attempting to match in step (C2), when said particular digital data item identifier obtained in step (C1) corresponds to an identifier in said first mapping data, using said first mapping data to determine a digital part identifier of each part comprising the particular data item;

> (C4) using said second mapping data and at least one digital part identifier determined in step (C3) to determine location data that identifies which of the plurality of servers in the file system stores the corresponding at least one part of the particular data item;
>
> (C5) attempting to access at least one part of the particular data item at one or more servers identified in step (C4) as storing said at least one part.

(‘096 patent, col. 38, l. 36 - col. 39, l. 28; Ex. 1001.)

At first glance, claim 1 of the ‘096 patent (like claim 1 of the related ‘544 patent) may appear long and complicated. However, in reality, it is relatively simple. The method involves determining “data identifiers” for two data items and “part identifiers” for each of their respective parts; storing “mapping data” to map the data identifiers to the part identifiers, and to map the part identifiers to the locations on the network where they are stored; and attempting to access a data item using the identifiers and the mapping data. Using the letter notation in the claim itself as a guide, the data items each consist of multiple parts (limitations (A) and (B) of claim), and the parts are replicated on multiple servers (limitation (A3)). Each part has a corresponding “part identifier” that is based, at least in part, on a hash of the bits in the part (limitation (A1)), and each data item has a “data item identifier" that is based, at least in part, on a hash of the contents of the data item (limitation (A2)). As dependent claim 2 confirms, these "data item identifiers" can be based on a hash of the part identifiers (*i.e.*, a "hash of hashes"). The "first mapping data" maps the data identifier to the corresponding part identifiers (portion (A4)), and the "second mapping data" maps the part identifiers to location data that identifiers the servers on the system that store the parts (limitation (A5)). The identifiers and mapping data are used to attempt to access a particular data item (limitations (C) and (C1)-(C5)) by the following process: first, finding the mapping data for a given data item identifier (limitations (C1)-(C2)); then, using the "first mapping data" to obtain the part identifiers (limitations (C3)); finally, using the "second mapping data" to obtain one of the parts from a server (limitations (C4)-(C5)).

The applicants stated that they were entitled to these broad claims because "[i]n ***all*** of the prior data processing systems, the names or identifiers provided to identify data items . . . are ***always*** defined relative to a specific context," and "there is ***no direct relationship*** between the data names and the data item." ('096 patent, col. 2, l. 26-31, col. 2, ll. 39-40, emphasis added; Ex. 1001.)

These representations were simply wrong. Prior data processing systems ***did use*** identifiers based on the contents of a data item or its segments – and not the context or pathname – ***including*** identifiers based on a "hash of hashes." In fact, these techniques were old and widely used. This is not surprising. The concept of using a mathematical function to create a "fingerprint" or "signature" for a data item based on the content of the data item predates the '096 patent by decades. For example, IBM developed one of the first hash tables in the 1950s (*see*, *e.g.*, G. D. Knott, "Hashing functions," 18 The Computer Journal 265 (1975), at 274 (discussing "history of hashing"); Ex. 1011), and Professor Ron Rivest of MIT introduced the MD5 hash algorithm referenced in the '096 patent in the early 1990s. (*See, e.g.*, R. Rivest, "The MD5 Message-Digest Algorithm," Internet RFC 1321 (Apr. 1992); Ex. 1012.)

Moreover, Professor Ralph Merkle and others were partitioning data items into parts, calculating identifiers for the parts using a hash function, and then "hashing the hashes" to create top-level signatures (*i.e.*, identifiers for the data items as a whole) by the 1970s.[7] (*See* Merkle, U.S. Patent No 4,309,569, entitled

[7] The idea of partitioning data into smaller parts (e.g., "pages" or "blocks") has been known for decades. (*See, e.g.,* B. Lampson and R. Sproull, "An Open Operation System for a Single-User Machine," ACM Operating System Review (Dec. 1979) at 101 ("The system organizes long-term storage (on disk) into files,

"Method of Providing Digital Signatures," filed Sept. 5, 1979, at col. 2, ll. 54-67 and Figure 1 (describes calculating signatures for a "vector of data items" by calculating signatures for segmented portions of the vector using a hash function, then combining the signatures using the same hash function) ("Merkle"); Ex. 1031.) "Hash trees" – also known as "Merkle trees" – were well known in the field long before the '096 patent.

These hashing functions take as input the data contained in a file, a portion of a file, or other data item, and produce a much smaller-sized output value, commonly called a "hash," "hash value," "message digest" ("MD"), or "checksum." (*See, e.g.*, McGraw-Hill Dictionary of Scientific and Technical Terms, (4th ed., 1989), at 860; Ex. 1013; *see also* Kaliski, "A Survey of Encryption Standards," IEEE Micro (Dec. 1993), pp. 74–81, at 77; Ex. 1014.) For example, a file that is a million bytes (or even much larger) in size can be used as input to produce a hash value that is a mere 16 bytes in length. Because of the mathematical properties of the function, the odds that two different files will

each of which is a sequence of fixed-size pages " and "[t]he data bytes of the file are contained in pages 1 through n."); Ex. 1032; *see also* A. Tanenbaum, "Operating Systems Design and Implementation," Prentice-Hall (1987) at 256; Ex. 1033.)

produce the same 16 byte hash are extremely small: for example, with a 16 byte hash output, the odds that two randomly picked inputs have the same hash are 2^{-64}, or approximately one in sixteen billion billions. (Kaliski at 77; Ex. 1014.) Consequently, hashes are known as "signatures" or "fingerprints" because they identify data items with high reliability, just like signatures or fingerprints identify people with a high degree of certainty. (*See* McGregor and Mariani, "'Fingerprinting' – A Technique for File Identification and Maintenance," 12 Software Practice & Experience 1165 (1982), at 1165 ("fingerprinting" technique "produce[s] a quasi-unique identifier for a file, derived from that file's contents . . . [t]he idea is to provide an identifying feature for every file, which is intrinsically distinctive, and analogous (hopefully) to a human's fingerprint."); Ex. 1017.)[8]

[8] This reference was central to the rejection of EP counterpart application EP0826181A1 with claims having a central feature of content-based identifiers. (Annex to the communication, May 8, 2009; Ex. 1020.) The applicants amended the claims to emphasize a "licensing" limitation not found in the challenged claims (Reply to communication from the Examining Division, Nov. 18, 2009 at 4; Ex. 1021), but this too was found unpersuasive and the rejection was maintained by the EPO. (Annex to the communication, March 14, 2012 at 4; Ex. 1022) Following this rejection, the applicants withdrew the application from consideration.

Although the applicants suggested in their patent application that they were the first to utilize hash functions to identify data items for file management applications, others working in the field used them for the same purposes more than a decade before the '096 patent. For example, at least sixteen years before the '096 patent was filed, researchers were already using hash functions to determine whether two records were identical, and to eliminate duplicate records. (*See*, *e.g*., Babb, "Implementing a Relational Database by Means of Specialized Hardware," 4 ACM Transactions on Database Systems, 1, 2-4 (March 1979); Ex. 1034; Bitton and DeWitt, "Duplicate Record Elimination in Large Data Files," 8 ACM Transactions on Database Systems 255, 256 (commenting on the work of Babb); Ex. 1035; *see also* Rabin, "Fingerprinting by Random Polynomials," Center for Research in Computing Technology, Harvard University, Report TR-15-81 at 1 and 9 (1981); Ex. 1015; Manber, "Finding Similar Files in a Large File System," Department of Computer Science, University of Arizona, TR 93-33 at 3; Ex. 1016; McGregor and Mariani at 1165; Ex. 1017.)

Many other printed publications and patents disclose and use identifiers exactly like those described and claimed in the '096 patent, including "hashes of

(Closing of Application, June 14, 2012; Ex. 1023.)

hashes," for exactly the same purposes. These publications disclose identifiers that are location- and context-independent, that are determined using only the contents of a data item or a segment of a data item, and that are formed using identical algorithms to those mentioned in the '096 patent.

Browne: For example, researchers at the University of Tennessee and Bell Laboratories disclosed a system that created "location-independent file names" (or "LIFNs") to identify files on the Internet. (Browne at 3; Ex. 1002)). LIFNs – like the identifiers in the '096 patent – uniquely identified files based on their contents, not their locations. (Browne at 3; Ex. 1002; *compare* '096 patent, col. 33, ll. 28-30; Ex. 1001.) LIFN <signatures> were computed as "the ascii form of the MD5 signature of the file" – the same function identified in the '096 patent. (Browne at 6; Ex. 1002; *compare* '096 patent, col. 12, ll. 45-47 (using MD5 or SHA); Ex. 1001.)

Browne specifically addressed compound data items ("resources"), including multiple files meant to be used together, such as a software package of computer program files. (Browne at 2, 5; Ex. 1002; *compare* '096 patent, col. 39 ll. 61-67 (confirming "data item" can include "a software product"); Ex. 1001.) To handle these compound resources, each component (*e.g.*, each file) of the resource was assigned its own LIFN <signature>, computed as the MD5 hash of the contents of the component. (Browne at 5-6; Ex. 1002.) The LIFNs for the components were then grouped together in a sequence, and a LIFN <signature> was computed for the resource as a whole by computing an MD5 hash of the sequence of LIFNs for the individual components. (*Id*. at 6.) In other words, hashes were computed for each component file, each hash acting as an identifier for its associated file, and a "hash of hashes" was then computed for the package, acting as the identifier for the package as a whole.

To access one of the components of a resource, a client first mapped the LIFN for the resource (e.g., the software package) to the LIFNs for the resource's components. It did this by sending the LIFN for the resource to a "LIFN database" which in turn provided a "composite-parts-list for the resource" which contained a list of LIFNs for each of the resource's component parts. (*Id*. at 4-6.) Once the client had obtained the composite-parts-list, it could then select the desired component's LIFN from the list and submit it to a "LIFN-to-location mapping service," which processed location queries and provided a second mapping to identify and return a list of server locations on the network that stored the component file so the client could then access it. (*Id*.) Consequently, this process of first mapping the LIFN for the resource (the "hash of hashes") to the LIFNs for its component parts (hashes of the parts), and then mapping the LIFN for a desired component (the hash) to one or more locations on the network that stored the component file, anticipated the very approach disclosed and claimed in the '096 patent. (*See* '096 patent, col. 8, ll. 40-42; col. 23, ll. 25-26, col. 33, ll. 28-30; *see also* col. 15, ll. 44-46; Ex. 1001.)

Langer in view of Satyanarayanan II: Another researcher, Albert Langer, also addressed the same problem as the '096 patent and, like Browne, proposed essentially the same solution. (Langer; Ex. 1003.) Langer was particularly concerned with sharing content on the Internet prior to the rise of the World Wide Web, through the use of popular protocols such as the File Transfer Protocol (FTP). FTP sites, among other things, could be accessed to provide a listing of available files at the site, and so a user could select and download files from the site. (*See, e.g.*, P. Deutsch et al., "How to Use Anonymous FTP," Internet RFC 1635 (May 1994); Ex. 1041.) Langer specifically addressed the problem of "uniquely identifying files which may have different names and/or be in different directories on different systems," and like the '096 applicants, observed that traditional location-based identifiers do not work well for distributed systems. (Langer at 3; Ex. 1003; *compare* '096 patent, col. 2, ll. 44-53; Ex. 1001.) Langer's solution, like the '096 patent, was to "provide a unique identifier for each file which is independent of location." (Langer at 3; Ex. 1003; *compare* '096 patent, col. 3, ll. 52-58; Ex. 1001.) Specifically, Langer disclosed "defining a unique identifier that does NOT include a particular site identifier," by "using a cryptographic hash function such as MD5," *i.e.*, the identical algorithm used in the '096 patent. (Langer at 4; Ex. 1003; *compare* '096 patent, col. 12, ll.45-47; Ex. 1001.)

Langer, like Browne, also addressed the issue of compound data items, including, for example, archived files that were part of the same package. (Langer at 5; Ex. 1003.) Langer extracted each file from the archive, and computed an identifier for it based on an MD5 hash of the contents of the file. (*Id.*) He then concatenated those identifiers together to create a new file (*i.e.*, a file of the sequence of MD5 hashes), and computed an MD5 hash of the contents of the new file (*i.e.*, a "hash of hashes") to serve as an identifier for the package as a whole. (*Id.*) A client could use the identifier for the package (*i.e.*, the hash of hashes) to obtain the file containing the sequence of MD5 hashes for the individual files, and then could select any of the MD5 hashes of the individual files to retrieve a particular item of interest from one of several remote FTP sites on the network that maintain copies of the item. (*Id.* at 3, 5.)

Langer specifically discussed the use of these unique identifiers for "mirror sites," noting that a client could be "automatically informed of the nearest location" from which a file could be downloaded. (*Id.* at 3–4.) Although Langer did not specifically discuss the details of these "mirror sites," maintaining multiple copies of files on different servers on a network was well known in the art. The '096 patent admits that file mirroring technology is old. ('096 patent, col. 3, ll. 16–29; Ex. 1001.) In fact, replicated file systems predated the '096 patent by decades. (*See, e.g.*, Alsberg and Day, "A Principle for Resilient Sharing of Distributed Resources," Center for Advanced Computation, Univ. of Illinois at Urbana-Champaign (1976), at 1 and 19; Ex. 1042; and Bartlett, "A 'NonStop' Operating System," 1978:3 Proceedings of the Eleventh Hawaii International Conference on System Sciences 103, 103, 112 (1978); Ex. 1043.) The *Coda* file system, for example, developed at Carnegie-Mellon University in the late 1980s, sought to improve file availability by maintaining multiple copies of each file on a network. (Satyanarayanan II at 448; Ex. 1028; *see also* M. Satyanarayanan, "Scalable, Secure, and Highly Available Distributed File Access," *IEEE Computer*, vol. 23, no. 5 (May 1990), pp. 9–21, at 13 (reviewing other contemporary distributed file systems) ("Satyanarayanan I"); Ex. 1005.) It would have been straightforward and obvious to utilize replicated file technology, such as taught by Satyanarayanan II among others, to provide more reliable storage for the files accessed and used by Langer. Satyanarayanan II provides express motivation to make this combination, by disclosing an efficient mechanism that provided a client with the nearest available copy of such a mirrored file. (*See* Satyanarayanan II at 450 ("[A] client obtains data from one member of its AVSG called the preferred server. The preferred server can be chosen . . . the basis of performance criteria such as physical proximity . . . ."); Ex 1028.) As further discussed *infra*, such a combination of Langer with Satyanarayanan II would have been the application of Satyanarayanan II's known techniques to the known device of Langer, ready for improvement, to yield the predictable result of improving access to files. (*See* Satyanarayanan II at 450; Ex. 1028.)

Kantor in view of Satyanarayanan II: Dr. Frederick W. Kantor, a physicist from Columbia University, developed yet another example of context- and location-independent identifiers for the same purposes as the '096 patent. Dr. Kantor described a product called FWKCS that created "contents-signatures" for files based on their content. (Kantor at Preface 2; Ex. 1004.)[9] FWKCS used these contents-signatures to uniquely identify files on a bulletin board system ("BBS"),

[9] The three-page Preface section of Kantor's FWKCS user manual does not have individual page numbers. Citations to the Preface are labeled "Preface" to denote pages 1-3 of the Preface section. Otherwise, citations refer to the page numbers in the top-right margin of the remainder of the user manual.

(*id.* at Preface 2), an online file system considered a precursor to the World Wide Web. The contents-signature, as the name suggests, was based on a hash of the data contained in a file, just like the identifiers in the '096 patent. (*Id.* at 8; Ex. 1004; *compare* '096 patent, col. 13, ll. 34-42 and Figure 10A; Ex. 1001).

Kantor, like Browne and Langer, also specifically addressed the issue of compound data items, in particular, "zipfiles" containing a set of other files meant to be kept together. (Kantor at Preface 1; Ex. 1004.) FWKCS created "zipfile contents-signatures" for these zipfiles, based on a hash of the contents-signatures of the files within the zipfile (*i.e.,* a "hash of hashes"), once again, just as in the '096 patent. (*Id.* at 9; Ex. 1004; *compare* '096 patent, col. 13, ll. 43-61 and Figure 10B; Ex. 1001.) FWKCS's contents-signatures accordingly could be used both to identify compound data items, like zipfiles, and to separately identify the parts (*i.e*., individual files) contained within them. (Kantor at Preface 2; Ex. 1004.)

While Kantor addressed the unique identification of files stored by BBS systems, he did not specifically address the underlying storage system because that was a concern for the BBS, not Kantor's system. However, the *Coda* system, among others, provided network-based file replication, and it would have been straightforward and obvious to use replicated storage, as taught by Coda and others, to provide more reliable storage for the BBS's files. For similar reasons as described above in conjunction with Langer, a person of ordinary skill in the art would have found it obvious to apply the teachings of Satyanarayanan II to Kantor (e.g., to increase the reliability and response time of requests for files stored by BBS systems).

These prior art references provide just a handful of many examples of the use of content-based identifiers, including "hashes of hashes," to perform basic file management functions. Indeed, the application of hash-based identifiers to these functions was so obvious that at least one commentator not only described the applications as "easy" but also posted these ideas publicly "to impede anyone who might independently have had the idea from patenting it." (Williams, "An algorithm for matching text (possibly original)," posted to the "comp.compression" newsgroup on January 27, 1992; Ex. 1037; *see also* R. Williams, "An Introduction to Digest Algorithms," Rocksoft (Nov. 1994), at 13 (further describing potential uses for file management purposes of identifiers based on the hash of the contents of a block of data); Ex. 1038.)

In short, other than perhaps coining a new phrase – i.e., True Name – for a very old concept, there is absolutely nothing new disclosed or claimed in the '096 patent concerning the use of location-independent, content-based data identifiers.

VI. SPECIFIC GROUNDS FOR PETITION

Pursuant to Rule 42.104(b)(4)-(5) and Practice Guide Fed. Register Vol. 77, No. 27, page 6873 Petitioners have submitted claim charts in connection with this Petition (Exhibits 1029, 1030, 1036), from the pending litigation between Petitioner and PersonalWeb Technologies LLC. Those charts set forth Petitioners' position with respect to those references and demonstrate that the challenged claims are anticipated and/or unpatentable in view of each of them. Petitioner also submits herewith the Declaration of Dr. Douglas Clark (Ex. 1009), a Professor of Computer Science at Princeton University. Dr. Clark confirms that the charts identify representative subject matter in each reference that teaches each and every limitation of the challenged claims. He likewise confirms how each claim is anticipated or, at a minimum, rendered obvious by the prior art.

A. Grounds of Invalidity for Challenged Claims 1, 2, 81 and 83 based on Browne as a Primary Reference

Ground 1: Browne Anticipates Challenged Claims 1, 2, 81 and 83

Browne was not referenced or discussed by the examiner during prosecution of the '096 patent.[10] It is prior art under at least 35 U.S.C. § 102(a) and anticipates

[10] Browne is cited on the face of the '096 patent as one of the over 400 references. Browne played no role in the prosecution of the '096 patent.

each of claims 1, 2, 81 and 83 of the ‘096 patent.

Browne describes the Bulk File Distribution (“BFD”) package developed by researchers at the University of Tennessee and Bell Laboratories as part of an effort to make scientific software easily accessible over the Internet. (Browne at 1, 6; Ex. 1002.) The BFD package is based on the concept of a “virtual repository,” which is a distributed network of physical software repositories, each residing on a different file server. (*Id*. at 1-2.) Files are mirrored on multiple servers “to increase availability (e.g., if one site is unreachable, the software may be retrieved from a different site) and to prevent bottlenecks.” (*Id*. at 2.)

Like the ‘096 patent, Browne begins by discussing the shortcomings of context- or location-dependent file identifiers. At the time, a virtual repository could be implemented using a Uniform Resource Locator (URL) to identify each file. (*Id*.)[11] The authors identify several problems with the use of location-based identifiers, such as URLs, to access virtual software repositories. Among other things, URLs are inadequate for ensuring the consistency of a software repository.

[11] A URL is a character string, such as “http://www.netlib.org/index.html,” that can be used to specify a transfer protocol (“HTTP”), a location (“www.netlib.org”), and a file name (“index.html”). (*See, e.g*., T. Berners-Lee et al., “Uniform Resource Locators (URL),” Internet RFC 1738 (Dec. 1994); Ex. 1018.)

(*Id.*) Moreover, a URL can only identify a single location; if a virtual repository offers multiple copies of the same file, each copy must be given its own URL. (*Id.*)

In order to address these shortcomings, Browne adopts the same solution that would be later proposed in the '096 patent: associating a unique identifier with the ***contents*** of a file, rather than with the ***location*** of the file. Indeed, Browne even uses the same terminology as the patent, referring to its file names as "location independent." In the BFD package, the identifier is called a "Location Independent File Name," or LIFN. (Browne at 3; Ex. 1002; *compare* '096 patent, col. 3, ll. 55-58 ("the identity of a data item is ***independent*** of its name, origin, ***location*** . . . .") (emphasis added); Ex. 1001.)

In Browne's preferred approach, the LIFN is computed as the MD5 hash of the contents of a file. (Browne at 6; Ex. 1002.) The MD5 algorithm provides a substantially unique fingerprint, meaning that two files with identical content will always have the same MD5 fingerprint, even if they are located on different servers, and even if the server administrators give them different names.[12] "Once a

[12] The general syntax for the LIFN is "lifn:netlib:<signature>", referencing the file access protocol ("lifn," similar to the "http" protocol identifier in a URL), the

LIFN has been assigned to a particular sequence of bytes, that binding may not be changed." (*Id*. at 3.)

To access a file, a client computer sends a query to a "LIFN database," including the LIFN <signature> (*i.e.,* the MD5 hash) of the desired file to be accessed. (Browne at 4-5; Ex. 1002.) A "LIFN-to-location mapping service" processes the query and returns a list of one or more servers on the network that store a copy of the file associated with that LIFN <signature>. (Browne at 4-5; Ex. 1002.) To be clear, this mechanism is just like the True File Registry (TFR) of the '096 patent, which receives a content-based identifier (True Name) and provides a list of servers (source IDs) that store a copy of that file. ('096 patent, col. 33, ll. 36-38; Ex. 1001.)

Browne also addresses compound data items (*e.g*., related files meant to be used together) in the same manner as the '096 patent. Browne refers to these compound items as "resources," and specifically addresses the need to ensure consistency between them. (Browne at 2, 5-6; Ex. 1002.) To achieve that goal, Browne computes LIFNs for each of the components (*e.g*., files) that make up the resource, based on an MD5 hash of the component. (*Id*. at 6.) The identifiers are

server handling the request ("netlib"), and the unique MD5 hash used to identify a file[12] ("<signature>"). (Browne at 4, 6; Ex. 1002.)

then combined in a sequence to obtain a new file, called a "composite-parts-list," and a LIFN is computed for the composite-parts-list by performing an MD5 hash of the sequence of LIFNs for the individual components (*i.e.,* a "hash of hashes"). (*Id*.)

To access a compound resource (*e.g*., a software package) or to access a particular component within that resource, a client first maps the LIFN for the resource (e.g., the software package) to the LIFNs for the resource's component parts. It does this by sending the LIFN <signature> for the resource to the "LIFN database" which in turn identifies the location of a "composite-parts-list for the resource," containing the list of LIFNs for the resources' s component parts. (*Id*. at 4-6.) The LIFN <signature> for the resource (*i.e*., hash of hashes) is thus mapped (or what Browne calls "aliased") to the composite-parts-list (i.e., the hashes identifying the component parts of the resource). (*Id*. at 6.)

After this first mapping is obtained between the LIFN for the resource and the LIFNs for the component parts, a client may then select the LIFN <signature> for a desired component part from the composite-parts-list, and submit it to a "LIFN-to-location mapping service." This mapping service processes location queries, using the LIFN as a "key," and provides a second mapping to identify and return a list of server locations on the network that store the component file.

(Browne at 4-6; Ex. 1002.) Browne specifically discusses a prototype implementation in which the "LIFN database is a simple key/data database in which the unique keys are LIFNs [i.e., MD5 hashes]," and which responds to such queries with a list of network locations that store the file. (*Id*. at 6.) The client may then retrieve the component part from one of the servers. (*Id.* at 4, 6.)

As set forth in detail in the claim chart (Ex. 1030), and as confirmed by Dr. Clark (Clark Decl., ¶¶ 17–50; Ex. 1009), Browne anticipates each of claims 1, 2, 81 and 83 of the '096 patent. Dr. Clark confirms, for example, that Browne discloses determining "data identifiers" for data items (LIFN <signatures> for resources) and "part identifiers" for each of their respective parts (LIFN <signatures> for the component parts in the resources). (Clark Decl., ¶¶ 20, 23–25; Ex. 1009.) These LIFN <signatures> are computed as an MD5 hash of the contents of the data items and parts (in the case of the LIFN <signature> for the resources, a "hash of hashes.") (*Id.*) Dr. Clark further confirms that Browne discloses storing "mapping data" to map the data identifiers to the part identifiers (the LIFN database and the composite-parts-list), and to map the part identifiers to the locations on the network where they are stored (the LIFN-to-location mapping service mapping), and that this mapping data can be used to access the resource or any of its components parts. (Clark Decl., ¶¶ 28–36; Ex. 1009.)

To assist the PTAB, the Petitioner has provided, in Section VII below, a chart identifying, for each limitation of each challenged claim, the specific portions of Browne disclosing the limitation. The claim chart (Ex. 1030) and the Declaration of Dr. Clark (at ¶¶ 17–50; Ex.1009) further set forth Petitioner's position identifying where and how Browne anticipates the challenged claims.

B. Grounds of Invalidity for Challenged Claims 1, 2, 81 and 83 based on Langer as a Primary Reference

Ground 2: Langer combined with Satyanarayanan II Renders Obvious Challenged Claims 1, 2, 81 and 83

Langer was not referenced or discussed by the examiner during prosecution of the '096 patent.[13] It is prior art under at least 35 U.S.C. § 102(b) and anticipates each of claims 1, 2, 81, and 83 of the '096 patent.

Langer addresses the problem of distributing files over the Internet. Langer predates the advent of the World Wide Web, and therefore focuses on earlier file distribution technologies, notably the *File Transfer Protocol* (FTP) and the *Archie* and *WAIS* search engines. (Langer at 2; Ex. 1003.) Langer provided his contribution to the "alt.sources.d" and "comp.archives.admin" Usenet newsgroups. At the time, Usenet was one of the most effective channels for researchers to

[13] Like Browne, Langer is cited on the face of the '096 patent as one of over 400 references. Langer played no role in the prosecution of the '096 patent.

discuss current technical issues and to distribute research materials.

Like Browne, Langer recognizes the limitations inherent in the use of context- or location-based file identifiers, and the benefits of "uniquely identifying files which may have different names and/or be in different directories on different systems." (*Id*. at 3.) For example, identifiers that are tied to a physical server do not allow a user to select another site that is physically closer. (*Id*.)

Langer's solution is exactly the same as the '096 patent: determine a substantially unique identifier for each file based on the ***content*** of the file rather than its ***location***, and associate that file with the unique identifier (Langer at 3-4; Ex. 1003; *compare* '096 patent, col. 3, ll. 55-58; Ex. 1001.) Langer expressly recognizes that such an identifier may be calculated by performing a hash function on the contents of the file:

> A simple method of defining a unique identifier that does NOT include a particular site identifier would be to use a hash function on the entire contents of the file. . . . I would suggest using a cryptographic hash function such as MD5 which generates a 16 byte result.

(Langer at 4; Ex. 1003.) The '096 patent tracks Langer's solution (which predates it by almost four years) down to the choice of the same MD5 hash function.

Like Browne, Langer also specifically addresses compound data items, including, for example, archived files that are part of the same package. (*Id*. at 5.) Langer observes that such a package may be distributed in a variety of archive formats, and thus files may appear to be different even though they have identical content. (*Id*.)

To address these compound data items, Langer divides the packages into their component files, and computes a unique identifier for each component by performing an MD5 hash on the contents of the component. (*Id*.) He then concatenates these identifiers for the components together in a sequence to create a new file (*i.e.*, a file of the sequence of MD5 hashes), and performs another MD5 hash on the contents of the new file (*i.e.,* a "hash of hashes") to serve as an identifier for the package as a whole. (*Id*.) Once again, this is the same algorithm adopted years later by the '096 patent to compute True Names for "compound data items." ('096 patent, col. 13, ll. 43-61 and Fig. 10(b); Ex. 1001.)

Langer uses these unique identifiers with a central database server, such as the Archie and WAIS search engines, which includes the data to map MD5 hashes to physical locations. (Langer at 3-4; Ex. 1003.) Based on this infrastructure, a client computer can access a file using its identifier. For example, a user can query the Archie or WAIS search engines to find which FTP server holds a copy of a file with a specified MD5 hash. (Langer at 3-4; Ex. 1003; *see also, e.g.*, EARN Staff, "Guide to Network Resource Tools," Internet RFC 1580 (March 1994) at 23-26 (WAIS), 29-31, 36-37 (Archie); Ex. 1039.) The client computer can then access the file from one of the previously-identified file servers, again using the file's unique identifier.[14] Langer's central database server thus directly anticipates the True File Registry of the '096 patent, which provides a list of the locations, such as file servers, where a file with a given True Name is stored. ('096 patent, col. 33, ll. 36-38; Ex. 1001.)

The same mechanism is used to access compound data items, such as archived packages. (Langer at 5; Ex. 1003.) A client computer can use the MD5 identifier for the package (i.e., the "hash of hashes") to obtain the sequence of MD5 identifiers for the individual files, and then can use the MD5 identifiers for the component files to retrieve any particular file from remote FTP sites on the network. (*Id*. at 3-5.)

[14] Similarly to Browne, Langer proposes to alias MD5 signatures to actual file names on the server: "A simple ftp implementation would just hardlink every file available for ftp to a filename encoding of it's [sic] MD5 token. Users would then ftp the directory path and filename of the MD5 token and obtain the file." (Langer at 4; Ex. 1003.)

Langer specifically discusses the use of these unique identifiers for "mirror sites," noting that a client could be "automatically informed of the nearest location" from which a file could be downloaded. (*Id*. at 3-4.) As Langer notes, one problem with accessing files using identifiers tied to particular FTP sites is that "closer sites may have the same item." (*Id*. at 3.) While "[c]ache and mirror sites can partially solve [this] problem," one benefit of Langer's substantially unique and location-independent identifiers is that users can be ***forced*** to lookup the location of a given file. (*Id*. at 3-4.) In this way, users are "automatically informed of the nearest location" of the file (and thus avoid the "strong temptation to just take the easy way out, and not bother . . . to check where to obtain a file locally.") (*Id*. at 4.)

Although Langer did not specifically discuss the details of these "mirror sites," maintaining multiple copies of files on different servers on a network was well known in the art. For example, Mahadev Satyanarayanan, a computer scientist at Carnegie Mellon University, described one such system for replicating (i.e., mirroring) files and automatically informing users of the nearest location a given file one year prior to Langer's newsgroup contribution. (*See* Satyanarayanan II; Ex 1028.) The *Coda* system provided "[o]ne mechanism, *server replication*, [that] stores copies of a file at multiple servers." (Satyanarayanan II at 447; Ex

1028.) For a given set of files (called a "volume"), the "degree of replication and the identity of the replication sites are specified when a volume is created and are stored in a volume replication database." (Satyanarayanan II at 450; Ex 1028.) To service a request for a given file, *Coda* selected a "preferred server" from which to retrieve the file based on criteria such as "physical proximity, server load, or server CPU power." (Satyanarayanan II at 450; Ex 1028.) As Dr. Clark confirms, it would have been obvious to combine the *Coda* system with Langer because *Coda* satisfies Langer's desire to automatically inform users of the nearest location of files by providing access to replicated copies of files on the basis of server proximity. (Clark Decl., ¶ 57; Ex. 1009.) A person of ordinary skill in the art, exercising ordinary creativity, would have been motivated to combine accessing files using the substantially unique identifiers described by Langer with the replication and proximity-based file access described by the *Coda* system. (Clark Decl., ¶ 57; Ex. 1009.)

As set forth in detail in the claim chart (Ex. 1036), and as confirmed by Dr. Clark (Clark Decl., ¶¶ 51–79; Ex. 1009), Langer in combination with Satyanarayanan II renders obvious each of claims 1, 2, 81, and 83 of the '096 patent. Dr. Clark confirm, for example, that Langer discloses determining "data identifiers" for data items (an MD5 hash for a package) and "part identifiers" for each of their respective parts (MD5 hashes for the component parts in the package). (Clark Decl., ¶ 59; Ex. 1009.) These MD5 hashes are computed on the contents of the data items and parts (in the case of the MD5 hash for the package, a "hash of hashes" of the part identifiers). (Clark Decl., ¶ 59; Ex. 1009.) Dr. Clark further confirms that Browne discloses storing "mapping data" to map the data identifiers to the part identifiers (the central database stores data providing, for each MD5 hash of a package, the location of a file server from which the file containing the MD5 hashes of the component files can be found), and to map the part identifiers to the locations on the network where they are stored (the central database server also stores data providing the location of a file server from which a component file identified by an MD5 hash can be found), and that this mapping data can be used to access the resource or any of its components parts. (Clark Decl., ¶¶ 61–65; Ex. 1009.)

To assist the PTAB Petitioner has provided, in Section VII below, a chart identifying, for each limitation of each challenged claim, the specific portions of Langer combined with Satyanarayanan II that render obvious the limitation. The claim chart (Ex. 1036) and the Declaration of Dr. Clark (at ¶¶ 51–79; Ex.1009) further set forth Petitioner's position identifying where and how Langer combined with Satyanarayanan II renders obvious the challenged claims.

C. Grounds of Invalidity for Challenged Claims 1, 2, 81 and 83 based on Kantor as a Primary Reference

Ground 3: Kantor combined with Satyanarayanan II Renders Obvious Challenged Claims 1, 2, 81 and 83

Kantor was not cited to the USPTO and not considered during prosecution of the '096 patent. It is prior art under at least 35 U.S.C. § 102(b) and renders each of claims 1, 2, 81, and 83 of the '096 patent obvious.

Kantor is a published manual that describes a software program called the Frederick W. Kantor Contents-Signature System Version 1.22 ("FWKCS"). (Kantor at Title Page; Ex. 1004.) Like the '096 patent, Kantor addresses the shortcomings of context- or location-dependent file identifiers. (Kantor at Preface 1; Ex. 1004; *compare* '096 patent, col. 3, ll. 30-44; Ex. 1001) These include the "problem of duplicate files on electronic bulletin board systems" or BBSs.[15] (Kantor at Preface 1; Ex. 1004.) BBS users would unwittingly or intentionally upload files to a bulletin board, which the bulletin board already had. (*Id.*) Consequently, bulletin board operators "were paying for hardware to provide the

[15] Before the World Wide Web, computers "dialed into" a file server or network of servers where users could exchange files or other information by uploading or downloading files.

capacity for these spurious [duplicate] files, and spending many hours trying to find and delete them." (*Id.*)

Kantor, like Browne and Langer, uses the same solution that would be later proposed in the '096 patent: associating a unique identifier with the ***contents*** of a file, rather than with the ***location*** of the file. Kantor calls these identifiers "contents-signatures," and uses them to identify a file based only on the contents of a file, and not its name, location, or other characteristics. (Kantor at Preface 2; Ex. 1004; *compare* '096 patent, col. 3, ll. 30-44; Ex. 1001.) These signatures can be used for various purposes, including, for example, identifying duplicate content already stored on the BBS system (*see, e.g.*, Kantor at Preface 2-3; Ex. 1004), using a "Lookup" command to identify whether a file to be uploaded to a BBS is already present on the BBS (*id.* at 173), and using a "Precheck" command to generate a report identifying files which are present on the BBS system but not on the user's computer. (*Id.*)

FWKCS computes the contents-signature based on a function of the data in a file. (*See id.* at 7-8.) Specifically, the contents-signature is constructed with "the 32-bit CRC [cyclic redundancy check][16] of the file contents and the uncompressed

[16] As Dr. Clark confirms, a CRC is a well-known hash function that calculates a

file-length." (*Id.*) The CRC and the file length (*i.e.,* file size) are both a function of the data contained in the file, and two files with the same content necessarily have the same contents-signature. (*Id.*; Ex. 1004.) In fact, Kantor uses the same technique as in the '096 patent, creating a contents-signature with a hash and a length value. (*Id.* at 7-8; Ex. 1004; *compare* '096 patent, col. 13, ll. 34-42 and Figure 10A; Ex. 1001.) Each contents-signature is location-independent and independent of the file's pathname, location, or context.

Kantor also creates identifiers for compound data items in the same manner as the '096 patent. Kantor explains that BBS users often bundled files into "zipfile" format, a well-known format for organizing related files into a single compound file. (*See* Kantor at Preface 2; Ex. 1004.) FWKCS generates "zipfile contents-signatures" for these zipfiles by computing the contents-signatures for each of the individual files within the zipfile, then hashing these contents-signatures using an "addition modulo 2^32" hash[17] to create the zipfile contents-

value as a function of the file's contents. (Clark Decl., ¶ 87; Ex. 1009.)

[17] As Dr. Clark confirms, "addition modulo 2^32" is another well-known hash function that uses addition to calculate a value based on a file's contents. (Clark Decl., ¶ 88; Ex. 1009; *see also* G. D. Knott, "Hashing functions," 18 The Computer

signature for the zipfile as a whole (*i.e.,* a "hash of hashes"). (Kantor at 9; Ex. 1004.)

FWKCS provides many operations for working with these zipfile and file contents-signatures. For example, FWKCS computes the zipfile and file contents-signatures for all of the zipfiles and individual files in the system, and stores them in a master contents-signature list ("cs-list"), such as "CSLIST.SRT," which is similar to the "True Name Registry" of the '096 patent. (*Id.* at 18; Ex. 1004; *compare* '096 patent, col. 33, ll. 36-38; Ex. 1001). In addition, when uploading a zipfile, FWKCS can determine whether that zipfile already exists in the system using the zipfile contents-signature, and can determine whether individual component files of that zipfile already exist in the system, using the contents-signatures for the individual files. (Kantor at 9; Ex. 1004.) The zipfile and file contents-signatures also can be used to find zipfiles or files on the BBS, to delete

Journal (1975), no. 3, at 268 (describing "common elementary hashing functions" including addition functions); Ex. 1011.) By adding together the values of the contents identifiers for the individual files, Kantor ensures that "the resulting [zipfile contents identifier] does not depend on the names of the files, the dates of the files, [or] the order in which they appear in the zipfile . . . ." (Kantor at 9; Ex. 1004.)

duplicate zipfiles or files uploaded under different names, and to determine if files are contained in a larger zipfile or spread among different zipfiles. (*Id.*)

Although BBS clients typically connected to a BBS and requested files based on the file's name (*see, e.g.*, F. Clark et al., "PCBoard v15.0 Technical Reference Manual," Clark Development Corporation, 1993 at 332;[18] Ex. 1040; *see also* Kantor at Preface 1; Ex. 1004), a person of ordinary skill in the art would have found it obvious to modify the BBS commands, including the download and/or read commands, to permit identifying files based on contents-signatures or zipfile contents-signatures. (Clark Decl., ¶ 83; Ex. 1009.) Among other things, this would facilitate integrity checking by more precisely specifying the file of interest by its content, and thus improve accuracy. (Clark Decl., ¶ 83; Ex. 1009.) Kantor shows that such a modification would be easy to implement. For example, FWKCS already had utilized contents-signatures as parameters specified in certain user commands, such as the "Lookup" operation (*see* Kantor at 97 and 173; Ex. 1004), and it would have been straightforward to similarly allow download and read commands to identify a file by a contents-signature. (Clark Decl., ¶ 83; Ex.

[18] Clark Development Corporation released PCBoard v15.0 in August 1993. Page numbers were added to the documentation included in this release for Ex. 1040 to conform to 37 C.F.R. 42.63.

1009.) Moreover, it would be an easy matter for a user to obtain the contents-signatures for the files of interest. For example, the signatures could be shared among users. (Clark Decl., ¶ 83; Ex. 1009.) In addition, the signatures could be provided to a user by the BBS itself using FWKCS through the Precheck operation or an easily modified version of this operation. (Clark Decl., ¶ 83; Ex. 1009.) Kantor describes the Precheck operation as an FWKCS utility for identifying files, based on their contents-signatures, which exist on the BBS but which do not yet reside on a user's computer. (Kantor at 173; Ex. 1004.) Using Precheck, a user is provided a report and can then use contents-signatures from the report to request files of interest with the modified download command. (Clark Decl., ¶ 83; Ex. 1009.) In addition, Kantor provided contents-signatures to the user in response to Lookup commands in certain modes of operation. (Kantor at 96-97; Ex. 1004; Clark Decl., ¶ 83; Ex. 1009.)

Moreover, although Kantor did not specifically disclose storing copies of a given file on multiple servers and then accessing the file from a preferred server, the *Coda* system developed at Carnegie Mellon University confirms that just such a technique was well-known before the '096 patent. (*See* Satyanarayanan II; Ex. 1028.) The *Coda* system provided "[o]ne mechanism, *server replication*, [that] stores copies of a file at multiple servers." (Satyanarayanan II at 447; Ex. 1028.)

For a given set of files (called a "volume"), the "degree of replication and the identity of the replication sites are specified when a volume is created and are stored in a volume replication database." (Satyanarayanan II at 450; Ex. 1028.) Once files are replicated in this way, Satyanarayanan II discloses that "a client obtains data from [a] preferred server. The preferred server can be chosen . . .on the basis of performance criteria such as physical proximity, server load, or server CPU power." (Satyanarayanan II at 450; Ex. 1028.) As Dr. Clark confirms, it would have been obvious to combine Satyanarayanan II with Kantor because Satyanarayanan II teaches a method of accessing files with improved reliability and response time that is directly applicable to Kantor's BBS systems. (Clark Decl., ¶ 84; Ex. 1009.) A person of ordinary skill in the art, exercising ordinary creativity, would have been motivated to combine accessing files using the contents-signatures described by Kantor with the reliable replicated file system described by Satyanarayanan II. (Clark Decl., ¶ 84; Ex. 1009.)

As set forth in detail in the claim chart (Ex. 1029), and as confirmed by Dr. Clark (Clark Declaration, ¶¶ 80-111; Ex. 1009), Kantor in view of Satyanarayanan II renders obvious each of claims 1, 2, 81, and 83 of the '096 patent. Dr. Clark confirms, for example, that Kantor discloses determining "data identifiers" for data items (zipfile contents-signatures) and "part identifiers" for each of their respective parts (contents-signatures for the component parts in the zipfile). (Clark Decl., ¶¶ 86-88; Ex. 1009.) These content-signatures are computed as a hash of the contents of the data items and parts (in the case of the data items, an addition modulo 2^32 has of the part identifiers, and in the case of the parts, a CRC hash of the contents of the parts). Dr. Clark further confirms that Kantor discloses storing "mapping data" to map the data identifiers to the part identifiers (the cs-list files), that Kantor in view of Satyanarayanan II discloses "mapping data" to map the part identifiers to the locations on the network where they are stored (the volume replication database of Satyanarayanan II), and that this mapping data can be used, with obvious modifications to Kantor, to access a resource or any of its components parts. (Clark Decl., ¶¶ 90-93; Ex. 1009.)

To assist the PTAB, the Petitioner has provided, in Section VII below, a chart identifying, for each limitation of each challenged claim, the specific portions of Kantor combined with Satyanarayanan II that render obvious the limitations. The claim chart (Ex. 1029) and the Declaration of Dr. Clark (at ¶¶ 80-111; Ex. 1009) further set forth Petitioner's position identifying where and how Kantor combined with Satyanarayanan II renders obvious the challenged claims.

VII. Claim Chart

To assist the PTAB in understanding the invalidity of the challenged claims, the Petitioner has provided a chart below identifying, for each limitation of each challenged claim, the specific portions of the references that disclose the limitation. The claim charts (Ex. 1029, 1030, 1036) and the Declaration of Dr. Clark (Ex. 1009) further set forth Petitioner's position identifying where and how the references anticipate or render obvious the challenged claims.

	Claim Limitations	**Browne**	**Kantor + Satyanarayanan II**	**Langer + Satyanarayanan II**
[1a]	A computer-implemented . . .	Browne 1995 at 1-7, FIGS. 1-3.	Kantor at Preface 2; Kantor at 5.	Langer at 2-4.
[1b]	(A) adding a data item . . .	Browne 1995 at 1-2, 4-6; *see* elements 1[c]-[1g].	Kantor at Preface 2; Kantor at 11, 194-95; *see* element [1c].	Langer at 5; *see* elements [1c]-[1g].
[1c]	(A1) for each part . . .	Browne 1995 at 1-6, FIG. 2; *see* element 1[b].	Kantor at 2-3, 6-8, 10-11, 31, 48-49, 51, 55; *see* element [1b].	Langer at 2-5; *see* element [1b].
[1d]	(A2) determining, using a . . .	Browne 1995 at 5-6; *see* elements [1b]-[1c].	Kantor at Preface 2; Kantor at 5, 9, 51-55; *see* elements [1b]-[1c].	*See* elements [1b] and [1c].
[1e]	(A3) storing each part . . .	Browne 1995 at 1-5, FIG. 2; *see* elements [1a]-[1b].	Satyanarayanan II at 447-59; *see* elements [1a]-[1b]	Langer at 3-4; *see* elements [1a]-[1b]; Satyanarayanan II at 447-59.
[1f]	(A4) storing first mapping . . .	Browne 1995 at 2-6, FIG. 3; *see* element [1b].	Kantor at 18, 36, 42, 45, 52-54; *see* elements [1b]-[1c].	Langer at 2-5; *see* element [1b].
[1g]	(A5) storing second mapping . . .	*See* elements [1b] and [1f].	Kantor at 52-54; *see* elements [1e]-[1f].	*See* elements [1b] and [1f].

	Claim Limitations	**Browne**	**Kantor + Satyanarayanan II**	**Langer + Satyanarayanan II**
[1h]	(B) repeating step (A) for . . .	Browne 1995 at 1-2, 5-6; *see* elements [1b]-[1g].	Kantor at 11, 194-95, *see* element [1b].	Langer at 4-6; *see* elements [1b]-[1g].
[1i]	(C) attempting to access . . . by: (C1) obtaining a particular . . .	*See* element [1f].	Kantor Preface at 2; Kantor at 3, 7-9, 33, 48, 51, 55, 96-97.	*See* element [1f].
[1j]	(C2) attempting to match . . .	*See* element [1f].	*See* elements [1f] and [1i].	*See* element [1f].
[1k]	(C3) based at least . . .	*See* elements [1b] and [1f].	*See* elements [1f] and [1i].	*See* elements [1b] and [1f].
[1l]	(C4) using said second . . .	*See* elements [1b] and [1f].	*See* elements [1g] and [1i].	*See* elements [1b] and [1f].
[1m]	(C5) attempting to access . . .	*See* elements [1b] and [1f].	*See* elements [1g] and [1i].	*See* elements [1b] and [1f].
[2]	The method of claim 1 wherein the digital identifier . . .	*See* elements [1b]-[1c].	*See* elements [1b]-[1c].	*See* elements [1b]-[1c].
[81a]	A computer-implemented . . .	*See* element [1a].	*See* element [1a].	*See* element [1a].
[81b]	obtaining, at said . . .	*See* elements [1b], [1c], and [1e].	*See* elements [1b]-[1c], and [1e].	Satyanarayanan II at 447-59; *see* elements [1b]-[1c], and [1e].
[81c]	determining, using hardware . . .	*See* element [1f].	Kantor at 52-54; *see* elements [1b] and [1e].	*See* element [1f].
[81d]	where the records . . .	*See* elements [1b]-[1c].	*See* elements [1b]-[1c], and [81c].	*See* elements [1b]-[1c].
[81e]	based at least . . .	*See* elements [1b] and [1f].	*See* elements [1i]-[1m].	*See* elements [1b] and [1f].
[83a]	A computer implemented . . .	*See* elements [1a]-[1c].	*See* elements [1a]-[1c].	*See* elements [1a]-[1c].
[83b]	(A) receiving a digital . . .	*See* elements [1b]-[1c], and [1e].	*See* elements [1b]-[1c], and [1e].	Satyanarayanan II at 447-59; *see* elements [1b]-[1c], and [1e].

	Claim Limitations	**Browne**	**Kantor + Satyanarayanan II**	**Langer + Satyanarayanan II**
[83c]	(B) hardware in combination . . .	*See* elements [1b], [1c], and [1f].	Kantor at 52-54; *see* elements [1b], [1e]-[1g], and [81c]-[81d].	*See* elements [1b]-[1c], and [1f].
[83d]	said database comprising . . .	*See* elements [1b]-[1c].	Kantor at 52-54; *see* elements [1b], [1e]-[1g], and [81c]-[81d].	*See* elements [1b]-[1c].
[83e]	(C) based at least . . .	*See* elements [1b], [1c], and [1e].	*See* elements [1b], [1e]-[1g], and [81c]-[81e].	*See* elements [1b]-[1c], and [1e].
[83f]	(D) using at least . . .	*See* elements [1b] and [1f].	*See* elements [1i]-[1m], and [81e].	*See* elements [1b] and [1f].

VIII. CONCLUSION

Based on the foregoing, it is clear that claims 1, 2, 81, and 83 of the ‘096 Patent recite subject matter that is either anticipated or obvious. The art cited above was never considered by the original Patent Examiner, and if it had been the ‘096 patent would not have issued. The Petitioner requests institution of an *inter partes* review to cancel those claims.

Respectfully Submitted,

/David L. Cavanaugh/

David L. Cavanaugh, Reg. No. 36,476
WilmerHale
1875 Pennsylvania Avenue NW
Washington, DC 20006

CERTIFICATE OF SERVICE

I hereby certify that, on December 17, 2012, I caused a true and correct copy of the foregoing materials:

- Petition for *Inter Partes Review* of U.S. Patent No. 8,001,096
- Exhibits 1001-1045
- Fee Summary Page
- EMC Corp. Power of Attorney

to be served via Federal Express on the following attorney of record as listed on PAIR:

Davidson Berquist Jackson & Gowdey, LLP
Attn: Brian Siritzky, Ph.D.
4300 Wilson Blvd., 7th Floor
Arlington, Virginia 22203

/David L. Cavanaugh/
David L. Cavanaugh
Registration No. 36,476

Table of Exhibits for U. S. Patent 8,001,096 Petition for *Inter Partes* Review

Exhibit	Description
1001.	U.S. Patent No. 8,001,096
1002.	S. Browne et al., "Location-Independent Naming for Virtual Distributed Software Repositories," University of Tennessee Technical Report CS-95-278 (Feb. 1995)
1003.	Albert Langer, "Re: dl/describe (File descriptions)," post to the "alt.sources" newsgroup on August 7, 1991
1004.	Kantor, "The Frederick W. Kantor Contents-Signature System Version 1.22," FWKCS122.REF (August 10, 1993)
1005.	M. Satyanarayanan, "Scalable, Secure, and Highly Available Distributed File Access," IEEE Computer, vol. 23, no. 5 (May 1990) (Satyanarayanan I)
1006.	S. Browne et al., "Location-Independent Naming for Virtual Distributed Software Repositories," http://www.netlib.org/utk/papers/lifn/main.html (Nov. 11, 1994)
1007.	K. Moore et al., "An Architecture for Bulk File Distribution," Network Working Group Internet Draft (July 27, 1994)
1008.	Chart of Patent Family Members
1009.	Declaration of Dr. Douglas Clark a Professor of Computer Science at Princeton University
1010.	Banisar et al., The Third CPSR Cryptography and Privacy Conference at 509 (1993)
1011.	G.D. Knott, Hashing functions, The Computer Journal 18

	(1975), no. 3, p. 265
1012.	R. Rivest, "The MD5 Message-Digets Algorithm," Internet RFC 1321 (Apr. 1992)
1013.	McGraw-Hill Dictionary of Scientific and Technical Terms, (4th ed., 1989)
1014.	B. Kaliski, "A Survey of Encryption Standards, " IEEE Micro (Dec. 1993)
1015.	Rabin, Fingerprinting by Random Polynomials, Center for Research in Computing Technology, Harvard University, Report TR-15-81
1016.	U. Manber, "Finding Similar Files in a Large File System", University of Arizona Technical Report (1994)
1017.	D.R. McGregor and J.A. Mariani 'Fingerprinting' – A Technique for File Identification and Maintenance, Software Practice & Experience 1165 (1982)
1018.	T. Berners-Lee et al., "Uniform Resource Locators (URL)," Internet RFC 1738 (Dec. 1994)
1019.	U. S. Patent 6,415, 280 Prosecution History, Response (August 22, 2001)
1020.	EP Pub. No. EP0826181A1 Prosecution History, Annex to the communication dated May 8, 2009
1021.	EP Pub. No. EP0826181A1 Prosecution History, Reply to communication from the Examining Division dated November 18, 2009
1022.	EP Pub. No. EP0826181A1 Prosecution History, Annex to the communication dated March 14, 2012
1023.	EP Pub. No. EP0826181A1 Prosecution History, Closing of

	Application dated June 14, 2012
1024.	U.S. Patent 8,001,096 Prosecution History, Application as filed on October 31, 2007
1025.	U.S. Patent 8,001,096 Prosecution History, Office Action of June 4, 2010
1026.	U.S. Patent 8,001,096 Prosecution History, Response of November 23, 2010
1027.	U.S. Patent 8,001,096 Prosecution History, Supplemental Response of December 26, 2010
1028.	M. Satyanarayanan et al., “Coda: A Highly Available File System for a Distributed Workstation Environment,” IEEE Transactions on Computers, vol. 39, no. 4 (April 1990) (“Satyanarayanan II”)
1029.	Invalidity Claim Chart in view of FWKCS Contents – Signature System Version 1.22 (“Kantor”)
1030.	Invalidity Claim Chart in view of LIFN (“Browne”)
1031.	Merkle, U.S. Patent No 4,309,569, entitled “Method of Providing Digital Signatures,” filed Sept. 5, 1979
1032.	Lampson and Sproull “An Open Operating System for a Single-User Machine,” ACM (1979)
1033.	A. Tanenbaum, “Operating Systems: Design and Implementation”, Prentice Hall (1987)
1034.	Babb, Implementing a Relational Database by Means of Specialized Hardware, ACM Transactions on Database Systems, Vol. 4, No.1, at 2-4, March 1979
1035.	D. Bitton and D. DeWitt, “Duplicate Record Elimination in Large Data Files, ACM Transactions on Database Systems, Vol.

	8, No. 2, at 255 – 265 (June 1983)
1036.	Invalidity Claim Chart in view of Langer
1037.	R. Williams, "An algorithm for matching text (possibly original)", posted to the "comp.compression" newsgroup on January 27, 1992;
1038.	R. Williams, "An Introduction to Digest Algorithms," Rocksoft (Nov. 1994)
1039.	EARN Staff, "Guide to Network Resource Tools," Internet RFC 1580 (March 1994)
1040.	F. Clark et al., "PCBoard v15.0 Technical Reference Manual," Clark Development Corporation, 1993
1041.	P. Deutsch et al., "How to Use Anonymous FTP," Internet RFC 1635 (May 1994)
1042.	P. Alsberg and J. Day, "A Principal for Resilient Sharing of Distributed Resources", Proc. of the 2d International Conference on Systems Sciences
1043.	J. Bartlett, "A 'NonStop' System", Proc. of the Eleventh Hawaii International Conference on System Sciences (1978)
1044.	U.S. Patent 8,001,096 Prosecution History, Preliminary Amendment of April 12, 2010
1045.	U.S. Patent 8,001,096 Prosecution History, Notice of Allowance of April 22, 2011